Kubernetes-Based-Canary-Deployment – Report

Abstract

Modern software systems require safe and reliable updates. Traditional deployments risk downtime or errors affecting all users. **Canary deployment** reduces this risk by gradually rolling out new versions to a portion of users.

This project demonstrates a canary deployment using **Kubernetes** and **Istio**, deploying a Pomodoro Timer app (Node.js + React) in two versions. Traffic was split between them for controlled rollouts, reflecting **progressive delivery practices** used in cloud-native environments.

Introduction:

Deploying new application versions to all users simultaneously can cause major outages. Canary deployment mitigates this by routing a small portion of traffic to the new version (“canary”) while the majority remains on the sheadtable version.

In this project:

- **v1 (Stable)** – 80% traffic
- **v2 (Canary)** – 20% traffic

Using **Istio**, we controlled traffic routing, monitored performance, and ensured safe rollouts, simulating real-world progressive deployment strategies.

Tools Used:

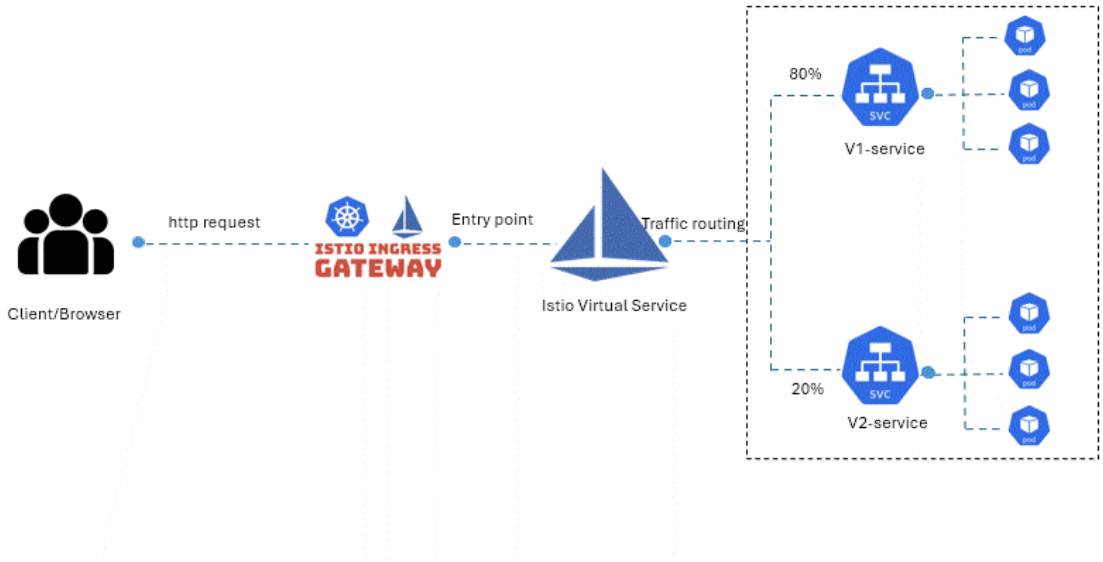
- **Kubernetes (MicroK8s / K3s):** Manages clusters, deployments, and services
- **Istio:** Enables traffic routing, observability, and safe rollouts
- **Docker:** Containerizes app versions for portability
- **Node.js + React (Vite):** Backend and frontend of the timer app
- **Nginx:** Serves static assets
- **Helm (optional):** Simplifies Istio installation
-

Steps Involved:

Canary-deployment

1. **Application Development:** Built timer app in Node.js + React (Vite), creating v1 and v2 builds.
2. **Containerization:** Docker images for each version ensured consistency.
3. **Cluster Setup:** MicroK8s/K3s Kubernetes cluster deployed.
4. **Deployment Manifests:** Kubernetes YAML files defined Deployments and Services for both versions.
5. **Istio Setup:** Installed Istio; configured Gateway, Virtual Service, and Destination Rule.
6. **Traffic Routing:** Split traffic **80% v1**, **20% v2**.
7. **Testing:** Verified routing via Istio Gateway; responses matched expected traffic split.
8. **Observability:** Monitored traffic and readiness using Istio metrics.
9.

Deployment Workflow Diagram:



Conclusion:

The Pomodoro Timer Canary Deployment project showcases a safe and efficient strategy for releasing application updates using Kubernetes and Istio. By employing canary deployments, new features can be gradually introduced to a subset of users, minimizing risk, ensuring stability, and maintaining an optimal user experience in production environments08-09-2025